Linear time temporal logic

Systems are often usefully modeled as **TRANSITION SYSTEMS**, where

STATES represent distinct states of system being modelled and
TRANSITIONS represent possible actual transitions between real-world states.

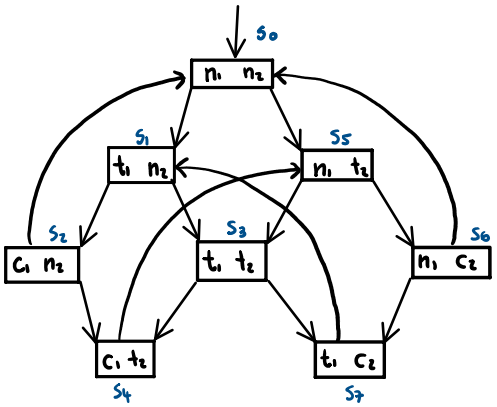
In addition we may have a number of basic **PROPERTIES** that may hold at states; eg.
- it is an error state
- flag 17 is set to true
- printer "Tiskaj" is busy ...

MOTIVATION EXAMPLE : MUTUAL EXCLUSION

IDEA:
- we have 2 parallel processes accessing a shared resource
- different processes are not allowed simultaneous access
- a process puts in a result if it wants to access the resource
- it uses the resource only once the request is granted
- once finished, the resource is released for other processes to use

We model this as a transition system:
Atoms = $\{n_1, n_2, t_1, t_2, c_1, c_2\}$

n = process is noncritical
(not accessing or trying to access resource)
t = is trying to access resource
c = is critical (it is accessing the resource)



LTL allows us to express useful properties of transition systems.
The basic idea is to express properties of infinite runs of the system.

Properties of above example:

- Mutual exclusion: the system is never in a state in which both process 1 and process 2 are critical simultaneously.
  In LTL: $G(\neg(c_1 \wedge c_2))$ This is an example of **SAFETY PROPERTY**: the system avoids some specified kind of undesirable state.

  Our model (obviously) satisfies mutual exclusion.

- If a process requests the resource then eventually the request is granted.
  In LTL: $G((t_1 \to Fc_1) \wedge (t_2 \to Fc_2))$

  Our model doesn't guarantee this. Eg. consider the run $s_0 \to s_1 \to s_5 \to s_7$

  This is an ex. of **LIVENESS PROPERTY**: Something good eventually hapenss.

DEF: A **TRANSITION SYSTEM** $M = (S, \to, L)$ is:
- S is a set (of states)
- $\to$ is a binary transition relation on S ($\to \subseteq S \times S$) satisfying $\forall s.\ \exists s'.\ s \to s'$
- L is a labeling function $L: S \to \mathcal{P}(Atoms)$; ie. L assigns to each state $s \in S$ a subset $L(s) \subseteq Atoms$ (Possibly one state is identified as a start state)

This says that we can't get stuck in a state (ie. there is a DEADLOCK)
Nonetheless we can model deadlock:

(we add a new atom d ... "deadlock")

LTL syntax

As with propositional logic, formulas are built over a collection of propositional atoms.

FORMULAS Φ, Ψ OF LTL ARE GIVEN BY:

- P - every prop. atom is a formula
- if Φ, Ψ are formulas, then so are $\top, \bot, \neg\Phi, \Phi \vee \Psi, \Phi \wedge \Psi, \Phi \to \Psi$ - so all formulas of prop. logic are formulas of LTL
- $X\Phi$ - Φ holds at the neXt state
- $F\Phi$ - Φ holds as some Future state
- $G\Phi$ - Φ holds Globally at all future states
- $\Phi U \Psi$ - Φ holds Untill (eventually) Ψ holds
- $\Phi W \Psi$ - Φ holds Weakly until Ψ
- $\Phi R \Psi$ - Φ Realeses Ψ

$\pi \models \Phi$: π satisfies Φ

The semantics of LTL is specified by defining a SATISFACTION RELATION of the form $\pi \models \Phi$, where π is a path through $\mathcal{M}$ and Φ is an LTL formula.

A PATH (or run) through $\mathcal{M}$ is an infinite sequence of transition-related states $\pi = s_1 \to s_2 \to s_3 \to \dots$

Notation: $\pi^i := s_i \to s_{i+1} \to s_{i+2} \to \dots$ "tail"

$\pi \models p \iff p \in L(s_i)$

$\pi \models \Phi \wedge \Psi \iff \pi \models \Phi$ and $\pi \models \Psi$ and similar for the other propositional connectives

$\pi \models X\Phi \iff \pi^2 \models \Phi$

$\pi \models F\Phi \iff \exists i \geq 1.\ \pi^i \models \Phi$

$\pi \models G\Phi \iff \forall i \geq 1.\ \pi^i \models \Phi$

$\pi \models \Phi U \Psi \iff \exists i \geq 1\ \ni:\ \pi^i \models \Psi$ and $\pi^j \models \Phi$ for all $j \in \{1, \dots, i-1\}$

$\pi \models \Phi W \Psi \iff$ either $\exists i \geq 1$ ... as for until ... or $\forall i \geq 1.\ \pi^i \models \Phi$ } $\Phi U \Psi \vee G\Phi$

$\pi \models \Phi R \Psi \iff$ either $\exists i \geq 1.\ \pi^i \models \Phi$ and $\forall j = 1, \dots, i.\ \pi^j \models \Psi$ or $\forall k \geq 1.\ \pi^k \models \Psi$ } $\neg(\neg\Phi\ U\ \neg\Psi)$

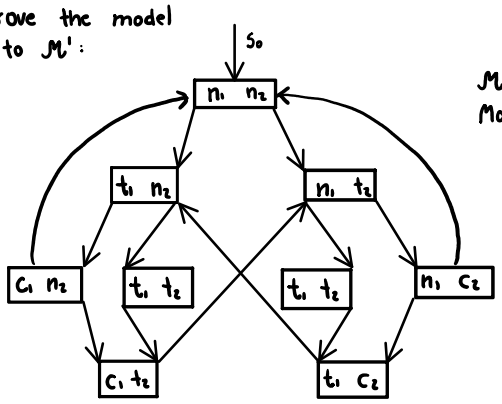
LTL model checking

$\mathcal{M}, s \models \Phi \iff \forall$ run $\pi = s_1 \to s_2 \to \ldots$ through $\mathcal{M}$ with $s_1 = s$, we have $\pi \models \Phi$
$\mathcal{M} \models \Phi \iff \forall$ run π through $\mathcal{M}$, $\pi \models \Phi$

Suppose we had designed earlier transition system as example protocol for mutual exclusion.
We could test the model for liveness by asking "is it the case that every run starting from s_0 satisfies $\Phi_{liveness} := G((t_1 \to F c_1) \wedge (t_2 \to F c_2))$?"
Equivalently does $\mathcal{M}, s_0 \models \Phi_{liveness}$; $\mathcal{M}$ our mutual exclusion model

We submit this to a MODEL CHECKER.
The model checker gives no and it gives a counter example run, eg. $s_0 \to s_1 \to s_6 \to s_7$.

Improve the model $\mathcal{M}$ to $\mathcal{M}'$:



$\mathcal{M}'$ does enjoy the property $\mathcal{M}', s_0 \models \Phi_{liveness}$ (dokaz dn)
Model checker for $\mathcal{M}'$ outputs yes.

An alternative to reducing the system: keep same system, but assume that it behaves in a fair way.
FAIRNESS PROPERTY:
$$\Phi_{fair} := GF(t_1 \wedge t_2) \longrightarrow (GF(t_1 \wedge t_2 \wedge X c_1) \wedge GF(t_1 \wedge t_2 \wedge X c_2))$$
To test a property Ψ of $\mathcal{M}$ under our fairness assumption we test $\mathcal{M}, s_0 \models \Phi_{fair} \to \Psi$.
In particular, it does hold $\mathcal{M}, s_0 \models \Phi_{fair} \to \Phi_{liveness}$!

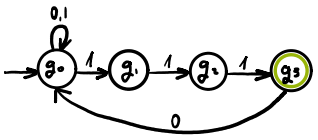
LTL MODEL CHECKING PROBLEM:
This is a nontrivial problem.

INPUT:
- transition system $\mathcal{M}$ (finite)
- start state $s \in \mathcal{M}$
- LTL formula Φ

OUTPUT: yes if $\mathcal{M}, s \models \Phi$
no ow, together with a path π through $\mathcal{M}$ starting at s, for which $\pi \not\models \Phi$ ($= \pi \models \neg\Phi$)

Nondeterministic Buchi automata

- A_1, $\Sigma_1 = \{0,1\}$



We run an NBA on an infinite word, eg.

0 1 1 1 0 0 1 1 1 1 0 0 0 1 1 1 1 1 0 1 1 ...

q_0 q_0 q_1 q_2 (q_3) q_0 q_0 q_0 q_0 q_1 (q_3) q_0 q_0 q_0 q_0 q_0 q_1 q_2 (q_3) q_0 ...

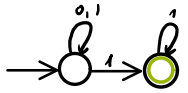
An NBA <u>accepts</u> an infinite word if there exists a run for that word which goes through a final state ∞-krat.

Eg. $(1110)^\omega := 1110111011\overline{10}\ldots$ is accepted

$1^\omega := 1111\ldots$ and $0^\omega := 000\ldots$ are not accepted.

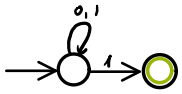
The ω-languge of the automata A_1 is $L\omega(A_1) = \{w \in \underbrace{\{0,1\}^\omega}_{\text{set of infinite sequences of 0s and 1s}} \mid w \text{ contains infinetly many substrings } 1110\}$

- A_2, $\Sigma = \{0,1\}$



$L\omega(A_2) = \{w \in \{0,1\}^\omega \mid w \text{ contains only finetly many 0s}\}$

- A_3



No ω-word has an accepting run in A_3: $L\omega(A_3) = \emptyset$

A **NONDETERMINISTIC BÜCHI AUTOMATA** over an alphabet Σ is (Q, I, Δ, F) where:

Q = finite set of states

$I \subseteq Q$ = set of initial states

$\Delta \subseteq Q \times \Sigma \times Q$ = set of transitions

$F \subseteq Q$ = set of final states

A ω-word $a_1a_2a_3\ldots \in \Sigma^\omega$ is **ACCEPTED** by the automata if there exists a path $q_0 \xrightarrow{a_1} q_1 \xrightarrow{a_2} q_2 \to \ldots$ in Δ such that $q_0 \in I$ and $q_i \in F$ for infinitely many i.

TESTING ON NBA FOR EMPTINESS

PROPOSITION: Let A be an NBA. Then $L\omega(A) \neq \emptyset$ iff there exists a path in A (a lasso) of the form $q_0 \to \ldots \to q_k \to \ldots$, where $q_0 \in I$, $q_k \in F$

ω-REGULAR LANGUAGES

We call sets $L \subseteq \Sigma^\omega$ ω-languages.

An ω-language is said to be ω-regular if there exists a NBA A $\exists$: $L = L\omega(A)$.

THEOREM: CLOSURE PROPERTIES OF ω-REGULAR LANGUAGES

If L_1 and L_2 are ω-regular then so are
- $L_1 \cup L_2$ (dn)
- $L_1 \cap L_2$ (dn, prove using GNBAs)
- $L^\omega - L_1$ (a deep result due to Büchi)

LTL MODEL CHECKING PROBLEM

We have a transition system M, LTL property Φ, some start state s_i.

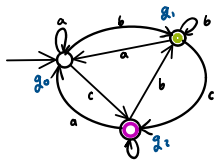
Question: Does $s_i \models \Phi$? (Does every infinite run π starting from s_i satisfy Φ?) ⇝ <u>Yes</u> or <u>No</u>, together with a lasso π for which $\pi \not\models \Phi$.

Generalized NBAs

GENERALISED NBA is $(Q, I, \Delta, \{F_j\}_{1 \le j \le k})$, where $\forall 1 \le j \le k,\ k \ge 1.\ F_j \subseteq Q$.

An ω-word $a_1 a_2 a_3 \ldots$ is accepted if $\exists$ run $q_0 \xrightarrow{a_1} q_1 \xrightarrow{a_2} q_2 \to \ldots$ with $q_0 \in I$ and $\forall j.\ 1 \le j \le k:\ q_i \in F_j$ for ∞-ly many i.

- A_4
  $\Sigma = \{a, b, c\}$



$k = 2$
$F_1 = \{q_1\}$, $F_2 = \{q_2\}$

$L_\omega(A_4) = \{w \in \{a,b,c\}^\omega \mid w$ contains only many b's and only many c's$\}$

PROPOSITION: TFAE for $L \subseteq \Sigma^\omega$.
1.) $\exists$ NBA A s.t. $L = L_\omega(A)$
2.) $\exists$ GNBA A s.t. $L = L_\omega(A)$

PROOF: 1 ⇒ 2 easy, 2 ⇒ 1 dn

TRACE LANGUAGE OF AN LTL FORMULA
Consider two runs (paths) $\pi = s_1 \to s_2 \to s_3 \to \ldots$ in $\mathcal{M}$
$\pi' = s_1' \to s_2' \to s_3' \to \ldots$ in $\mathcal{M}'$
two (potentially) different transition systems

Suppose also that $\forall n.\ L(s_n) = L(s_n')$. Then it follows that $\forall$ LTL formulas Φ: $\pi^n \models \Phi \iff (\pi')^n \models \Phi \quad \forall n$.
So whether or not a path π satisfies a formula Φ depends only on the TRACE of the path.
trace $(\pi) := n \mapsto L(s_n)$
Or equivalently, we view trace (π) ω as ω-word over alphabet $\mathcal{P}(Atoms)$: $L(s_1)\ L(s_2)\ L(s_3)\ \ldots$

$Traces(\Phi) := \{A_1 A_2 A_3 \ldots \mid A_1 A_2 A_3 \ldots$ can arrise as trace (π) for some π s.t. $\pi \models \Phi\}$

Given Φ, we define a GNBA A_Φ s.t. $\boxed{L_\omega(A_\Phi) = Traces(\Phi)}$ ⇝ so language $Traces(\Phi)$ is ω-regular

To simplify matters, assume Φ uses only $\neg, \vee, \top, X, U$. Our goal is to construct A_Φ.
DN: show that LTL formula is equivalent to one involving the above connectives only.
$F\Phi = \top U \Phi$, $G\Phi = \neg F \neg \Phi$

States will be certain sets of formulas.
COMPLEMENTED SUBFORMULAS of Φ are defined by:
- $\Phi \in cs(\Phi)$
- $\Psi \in cs(\Phi)$ and it is not a negation ⇒ $\neg\Psi \in cs(\Phi)$
- $\neg\Psi \in cs(\Phi)$ ⇒ $\Psi \in cs(\Phi)$
- $X\Psi \in cs(\Phi)$ ⇒ $\Psi \in cs(\Phi)$
- $\Psi \vee \Psi' \in cs(\Phi)$ or $\Psi U \Psi' \in cs(\Phi)$ ⇒ $\Psi, \Psi' \in cs(\Phi)$

Ex. $cs(aUb) = \{aUb,\ \neg(aUb),\ a,\ b,\ \neg a,\ \neg b\}$

A subset $S \subseteq cs(\Phi)$ is **ELEMENTARY** if
- $\forall \neg\Psi \in cs(\Phi)$ either $\neg\Psi \in S$ or $\Psi \in S$, but not both
- $\Psi \vee \Psi' \in S$ ⇒ $\Psi \in S$ or $\Psi' \in S$
- if $\top \in cs(\Phi)$ ⇒ $\top \in S$
- $\Psi' \in S$ and $\Psi U \Psi' \in cs(\Phi)$ ⇒ $\Psi U \Psi' \in S$
- $\Psi U \Psi' \in S$ and $\Psi' \notin S$ ⇒ $\Psi \in S$

Ex. elementary subsets $cs(aUb)$ are: $\{a, b, aUb\}$
$\{a, \neg b, aUb\}$
$\{a, \neg b, \neg(aUb)\}$
$\{\neg a, b, aUb\}$
$\{\neg a, \neg b, \neg(aUb)\}$

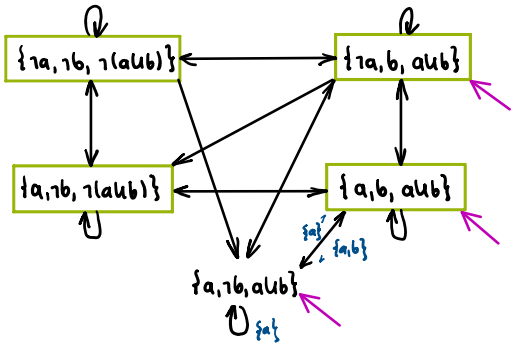
States of GNBA are elementary subsets of $cs(\Phi)$

$S \xrightarrow{A} S' \iff A = S \cap Atoms$ and $X\Psi \in S \Rightarrow \Psi \in S'$
and $\neg X\Psi \in S \Rightarrow \neg\Psi \in S'$
and $(\Psi U \Psi' \in S$ and $\Psi' \in S) \Rightarrow \Psi U \Psi' \in S'$
and $(\neg(\Psi U \Psi') \in S$ and $\Psi \in S) \Rightarrow \neg(\Psi U \Psi') \in S'$

$I = \{S \mid \Phi \in S\}$

Let $\psi_1 U \psi_1', \dots, \psi_k U \psi_k'$ be all until formulas in $cs(\phi)$
For every such formula $\psi_j U \psi_j'$ $(1 \le j \le k)$: $F_j = \{S \mid \psi_j' \in S \text{ or } \neg(\psi_j U \psi_j') \in S\}$

$\phi = a U b$:



In general we get a GNBA
In our example we obtain an NBA

LTL MODEL CHECKING - GENERAL ALGORITHM

INPUT: $\mathcal{M}, s, \Phi$
QUESTION: does $\mathcal{M}, s \models \Phi$ hold?

STEP 1 : We construct GNBA $\mathcal{A}_{\neg\phi}$. Convert $\mathcal{A}_{\neg\phi}$ to an equivalent NBA $\mathcal{A}'_{\neg\phi}$ (recognises the same language)
An accepting run in $\mathcal{A}'_{\neg\phi}$ corresponds to a trace $A_0A_1\dots$: $A_0A_1\dots \notin Traces(\phi)$

STEP 2 : Construct a PRODUCT NBA $\mathcal{A}'_{\neg\phi} \otimes \mathcal{M}$.
Accepting runs in the product NBA correspond to paths through $\mathcal{M}$ starting from s along which $\neg\phi$ holds, ie. paths π from s : $\pi \models \neg\phi$.

STEP 3: Test whether $\mathcal{L}_\omega(\mathcal{A}'_{\neg\phi} \otimes \mathcal{M}) = \emptyset$ (use emptiness testing for NBAs)
- if it is empty → return yes , $\mathcal{M}, s \models \phi$
- if it is not empty → return no , use the lasso path given by emptiness test to extract a lasso $s = s_0 \to s_1 \to \dots \to s_k \to \dots$ in $\mathcal{M}$.

This lasso path π satisfies $\pi \models \neg\phi$.